Svi najkraći putevi

Ulaz

G = (V, E) - usmeren ili neusmeren težinski graf sa nenegativnim težinama grana

Izlaz

Dužine najkraćih puteva između svaka dva čvora.

PODSEĆANJE: Proširimo (na prethodnom času opisani) problem nalaženjem najkraćih puteva između svaka dva grada. Opštije, za dati orijentisani graf *G=(V,E)* sa nenegativnim težinama grana odrediti najkraće puteve između svih parova čvorova. Najpoznatiji algoritam koji rešava ovaj problem **je Flojd-Varšalov** algoritam čiji opis sledi.

Rešenje

- Prva ideja:
  Indukcija po broju grana.
  Prilikom dodavanja nove grane (u, w) može doći do promene dužine puta između bilo koja dva čvora v_1 i v_2. Proveravanje zahteva proveru d(v_1, u) + d(u, w) + d(w, v_2). Složenost: O(|E||V|²) => O(|V|⁴)

- Druga ideja:
  Indukcija po broju čvorova.
  Prilikom dodavanja novog čvora u, moramo prvo za svaki čvor v da proverimo d(u, v) i d(v, u). Dalje, tražimo najkraće puteve od u do ostalih čvorova (tražimo susede i proveravamo koji je put najkraći), kao i obratno, od ostalih čvorova do u. Ali to nije dovoljno. Potrebno je proveriti da li je neki od dosadašnjih najkraćih puteva moguće zameniti nekim koji sadrži u. Složenost: O(|V|³)

Finalna ideja: Ograničavamo dozvoljene puteve, i radimo indukciju po tom ograničenju, a ne menjamo skup čvorova i grana.

Numerišemo čvorove grafa od 1 do |V|. Puteve čiji čvorovi (ne računamo početak i kraj puta) imaju indekse manje ili jednake od k nazivamo k-put.

Baza indukcije: Za m = 1, razmatraju se samo direktne grane, rešenje je očigledno.

Induktivna hipoteza: Umemo da odredimo dužine najkraćih puteva između svaka dva čvora, pri čemu su dozvoljeni samo k-putevi za k < m.

Želimo da proširimo i za k = m.

Tražimo samo najkraće m-puteve i proveravamo da li oni poboljšavaju k-puteve za k < m. Za put od u do v je potrebno da proverimo da li je najkraći (m-1)-put od udo v_m + najkraći (m-1)-put od v_m do w manji od najkraćeg (m-1)-puta od udo w.

Algoritam je brži od prethodnog za konstantni faktor (isto je $O(|V|^3)$), i lakši je za implementaciju.

Algoritam

```
for (m : 1 .. n) {
    for (i : 1 .. n) {
        for (j : 1 .. n) {
            W[i, j] = min(W[i, j], W[i, m] + W[m, j]);
        }
    }
}
```

Zadaci

Zadatak 1

Granama grafa G=({1,2,3,4}, E) su pridružene težine kao u tabeli:

gran a	težin a
(1,2)	2
(1,3)	8
(2,3)	3
(2,4)	4
(3,4)	7
(4,1)	5

Odrediti sve najkraće puteve (all shortest paths) između čvorova ovog grafa.
REŠENJE: Matrica W^i se formira prema algoritmu izloženom i knjizi u odeljku 6.7
Algoritam SviNajkraciPutevi (W)
{ for (m=0; m<|V|; m++)
for(x=0; x<|V|; x++)
for(y=0;y<|V|; y++)
if (W[x,m] +W[m,y] <W[x,y]) W[x,y]= W[x,m] +W[m,y] ; /*nadjen je kraci put od x do y od predjasnje postavljene duzine*/
}

W^0	1	2	3	4
1	0	2	8	beskonačno
2	beskonačno	0	3	4
3	beskonačno	beskonačno	0	7
4	5	beskonačno	beskonačno	0

W^1	1	2	3	4
1	0	2	8	beskonačno
2	beskonačno	0	3	4
3	beskonačno	beskonačno	0	7
4	5	7	13	0

7=5+2 13=5+8

W^2	1	2	3	4
1	0	2	5	6
2	beskonačno	0	3	4
3	beskonačno	beskonačno	0	7
4	5	7	10	0

5=2+3 6=2+4 10=7+3

W^3	1	2	3	4
1	0	2	5	6
2	beskonačno	0	3	4
3	beskonačno	beskonačno	0	7
4	5	7	10	0

W^4	1	2	3	4
1	0	2	5	6
2	9	0	3	4
3	12	14	0	7
4	5	7	10	0

9=4+5 12=7+5 14=7+7

Zadatak 2

Zadat je težinski graf iz prethodnog zadatka sa nenegativnim težinama grana. Naći središte grafa.

REŠENJE:Središte grafa je čvor *v* takav da ima najamanju ekscentričnost. Ekscentričnost čvora *v* je maksimum najkraćih rastojanja od svih čvorova grafa G do čvora *v*, tj.
ecc(*v*) = max { W[*i*, *v*] | za svaki čvor *i* }

Koraci:

1. **primenom algoritma Svi_najkraci_putevi se nadje matrica W najkracih rastojanja izmedju svih parova cvorova**
2. **nadju se maksimumi po kolonama matrice W**
3. **nadje se vrednost minimuma ovih maksimuma i proglasiti za srediste grafa onaj cvor kojem odgovara ta vrednost**
   U konkretnom slučaju grafa iz prethodnog zadatka je pronadjena matrica W u četvrtoj iteraciji:

$W^{4}=W$	1	2	3	4
1	0	2	5	6
2	9	0	3	4
3	12	14	0	7
4	5	7	10	0

Maksimumi po kolonama su:

- **ecc(1) = 12**
- **ecc(2) = 14**
- **ecc(3) = 10**
- **ecc(4) = 7**

Odavde se zaključuje da središte grafa je čvor 4, zato što on ima najmanju ekscentričnost (čija vrednost jeste 7).

Zadatak 3

Zadat je težinski graf *G=(V,E)* sa nenegativnim težinama grana. Za zadata dva čvora *i,j* skupa *V* pronaći put minimalne dužine (a ne samo težinu tog puta).

REŠENJE: Bez smanjenje opštosti, indeksiramo vrste i kolone od 1..n umesto od 0 do n-1.
Izmena algoritma Svi_Najkr_putevi (glava 6.7 knjige) , tako da se formira matrica prethodnika *P* gde element *P[i,j]* pamti čvor koji je neposredni prethodnik čvora *j* na najkraćem putu od čvora *i*.
Jasno je da:
1. P[i,j]=0 ako je i==j ili ako W[i,j]=beskonacno
2. P[i,j]=i ako i !=j i ako w[i,j] < beskonacno
Dakle, najpre se algortmom Svi_najkraci_putevi u tri for ciklusa ažurira polazna matrica težina W, tako da na kraju čuva samo najkraće težine (rastojanja) između dva čvora. Tokom ažuriranja matrice W, vrši se i formiranje matrice čvorova P, tako što:

ako je u k iteracija spoljašnjeg for ciklusa od svih puteva medju čvorovima i,j koji prolaze kroz međučvorove 1,2,..k nađen trenutno najkraći put, onda:

1. ako se čvor k ne nalazi na tom putu, onda su svi medjučvorovi iz skupa {1,2,..,k-1}, pa je to ista ocena najkraćeg puta dobijena iz prethodne (k-1).ve iteracije sa istim prethodnikom P[i,j]
2. ako se čvor k nalazi na tom putu od i do j, onda ovaj put može da se podeli na puteve od i do k, i na put od k do j. Put izmedju čvorova i, k prolazi kroz medjučvorove **1..k-1** i on je deo najkraćeg puta od **i** do **j** u **k**-toj iteraciji, pa je kao u dokazu sa predavanja to i najkraci put od **i** do **k**. Slično se pookazuje da je put

od **k** do **j** najkraći put od **k** do **j** koji prolazi kroz čvorove **1**..**k-1**.
Dakle, kako je drugi segment puta zajednički, onda prethodnik u **k**-toj iteraciji **P[i,j]** čvora **j** na najkracem putu je istovetan kao i prethodnik **P[k,j]** iz (k-1)-ve iteracije
Prema tome, najkraći put od čvora **i** do čvora **j** kroz čvorove **1..k** se dobija kao manji od najkraćeg puta između **i,j** kroz medjučvorove **1,..., k-1** i zbira najkraćeg puta između čvorova **i,k** i puta izmedju **k,j** kroz čvorove 1..k-1 sto i jeste u jezgru najugnjezdenijeg for ciklusa.

Potom se od dobijene matrice P rekonstruiše najkraći put u proceduri putanja.

```
Algoritam Svi_najkraci_putevi (W)
Ulaz:W(matrica tezina)
Izlaz: W(matrica duzina najkracih puteva), P(matrica prethodecih cvorova na
najkracim putevima)
{
for (k=1; k<=n; k++)
  for (i=1; i<=n; i++)
   for (j=1; j<=n; j++)
     if (  W[i,j]>W[i,k]+W[k,j]  )
       {  P[i,j]=P[k,j];
          W[i,j]=W[i,k] + W[k,j]
      }
}
/*iz matrice P rekonstruisemo najkraci put izmedju dva zadata cvora  i,j */
Algoritam PUTANJA(i,j)
{
    if (i==j)
      stampati (i);  /*povratak*/
    else
       if  (P[i,j]==0) stampati "(Nema puta izmedju i, j)";
       else
            {PUTANJA(i, P[i,j] );
            stampati (j);
            }
}
```

Inicijalno: $D = \begin{bmatrix} 0 & 2 & 8 & \infty \\ \infty & 0 & 3 & 4 \\ \infty & \infty & 0 & 7 \\ 5 & \infty & \infty & 0 \end{bmatrix}$ $P = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 \\ 4 & 0 & 0 & 0 \end{bmatrix}$

k=1: $D = \begin{bmatrix} 0 & 2 & 8 & \infty \\ \infty & 0 & 3 & 4 \\ \infty & \infty & 0 & 7 \\ 5 & \mathbf{7} & \mathbf{13} & 0 \end{bmatrix}$ $P = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 \\ 4 & \mathbf{1} & \mathbf{1} & 0 \end{bmatrix}$

k=2: $D = \begin{bmatrix} 0 & 2 & \mathbf{5} & \mathbf{6} \\ \infty & 0 & 3 & 4 \\ \infty & \infty & 0 & 7 \\ 5 & 7 & \mathbf{10} & 0 \end{bmatrix}$ $P = \begin{bmatrix} 0 & 1 & \mathbf{2} & \mathbf{2} \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 \\ 4 & 1 & \mathbf{2} & 0 \end{bmatrix}$

k=3: $D = \begin{bmatrix} 0 & 2 & 5 & 6 \\ \infty & 0 & 3 & 4 \\ \infty & \infty & 0 & 7 \\ 5 & 7 & 10 & 0 \end{bmatrix}$ $P = \begin{bmatrix} 0 & 1 & 2 & 2 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 \\ 4 & 1 & 2 & 0 \end{bmatrix}$

k=4: $D = \begin{bmatrix} 0 & 2 & 5 & 6 \\ \mathbf{9} & 0 & 3 & 4 \\ \mathbf{12} & \mathbf{14} & 0 & 7 \\ 5 & 7 & 10 & 0 \end{bmatrix}$ $P = \begin{bmatrix} 0 & 1 & 2 & 2 \\ \mathbf{4} & 0 & 2 & 2 \\ \mathbf{4} & \mathbf{4} & 0 & 3 \\ 4 & 1 & 2 & 0 \end{bmatrix}$

Dakle, neka rezultujuća matrica puteva P izgleda:

0	1	2	2
4	0	2	2
4	4	0	3
4	1	2	0

Tada je:
prvi poziv npr. Putanja (1,4)
kako P[1,4] !=0, drugi poziv je Putanja (1,2), jer P[1,4]=2
kako P[1,2] !=0, drugi poziv je Putanja (1,1), jer P[1,2]=1
kako 1==1, stampa se 1
dalje, štampa se 2
dalje, štampa se 4
DAKLE, najkraći put izmedju čvorova 1 i 4 jeste dužine 6 (po zadatku 1), a ima putanju (((1),2),4)

Zadatak 4

E = { (a, b), (a, d), (a, c), (b, c), (c, d) }
3 4 2 9 1

m = 0 | a | b | c | d |
a	0	3	2	4
b	3	0	9	∞
c	2	9	0	1
d	4	∞	1	0

m = 1	a	b	c	d		m = 2	a	b	c	d
v = a	a	0	3	2	4	v = b	a	0	3	2	4
	b	3	0	[5]	[7]		b	3	0	5	7
	c	2	[5]	0	1		c	2	5	0	1
	d	4	[7]	1	0		d	4	7	1	0

m = 3	a	b	c	d		m = 4	a	b	c	d
v = c	a	0	3	2	[3]	v = d	a	0	3	2	3
	b	3	0	5	[6]		b	3	0	5	6
	c	2	5	0	1		c	2	5	0	1
	d	[3]	[6]	1	0		d	3	6	1	0

Zadatak 5

Modifikovati algoritam da u slučaju da postoje dva puta iste dužine, da bira onaj sa manjim brojem grana.

Da li je potrebno pamtiti puteve za ovo?

Nije, možemo svakoj grani dodati neku fiksnu težinu poput 1/|V| (ako su težine u grafu celobrojne), ili koristiti uređene parove (dužina, broj grana) i definisati poredak i funkciju minimum nad njima.

Zadatak 6

Dokazati da algoritam radi korektno i za težinske grafove koji imaju i negativne težine, pod uslovom da ne postoji ciklus negativne težine.

Rešenje:
Koristili smo pretpostavku da se na najkraćem $(m - 1)$-putu od u do v čvor v_m može naći samo jednom. Ova pretpostavka važi i u prisustvu negativnih težina, ali ne ako postoji ciklus negativne težine (u tom slučaju bismo dobijali sve kraće i kraće puteve što više puta prođemo ciklus)

Zadatak 7

$G = (V, E)$ - usmeren težinski graf, sa pozitivnim težinama.

Konstruisati algoritam za nalaženje dužine najkraćeg puta od čvora v do čvora w koji se sastoji od tačno k grana. Put ne mora da bude prost.

Rešenje: Indukcijom po k. Tražimo najkraći put od v do svih ostalih čvorova grafa. Označimo sa $w.sp\_k$ dužinu najkraceg puta od čvora v do čvora w koji sadrži tačno k čvorova.

Baza indukcije: težina direktne grane, ili ∞ u suprotnom.

Induktivna hipoteza: Znamo da nadjemo najkraiće puteve koji sadrže $k-1$ čvorova.

Induktivni korak: Postavljamo $w.sp\_k = \infty$ za svako w. Za svaku granu (x, y)postavljamo $y.sp\_k$ na minimum trenutne vrednosti i $x.sp\_k\text{-}1 + d(x, y)$.

Složenost: Imamo k iteracija, u svakoj iteraciji imamo $O(1)$ za svaki od čvorova i svaku od grana $O(k(|V | + |E|))$.

Zadatak 8

$G = (V, E)$ - težinski graf, sa nenegativnim težinama.

Konstruisati algoritam za određivanje ciklusa najmanje težine.

Rešenje - inicijalizovati elemente na dijagonali na ∞

Zadatak 9

Ako je ekscentricitet čvora i grafa G(V,E) maksimum najkraćih rastojanja od svih ostalih čvorova, napisati C program koji će odrediti čvor najmanjeg ekscentriciteta – središte grafa. Primena: određivanje optimalne lokacije, na primer, za stanicu hitne pomoći.

Uputstvo. Korišćenjem Flojdovog algoritma odrediti najkraća rastojanja između svih čvorova. Zatim, odrediti min ekscentriciteta svih čvorova grafa.

Zadatak 10

Dato je **n** kutija dimenzija (*xi*, *yi*, *zi*). Odrediti maksimalni broj kutija koje se mogu staviti jedna u drugu.

Uputstvo:
Formirati matricu veze A u kojoj je $ai, j = -1$, ako se u i-tu kutiju može smestiti kutija j, u protivnom $a\ i, j = \infty$ (*MAX*).
Zatim, primenom Flojdovog algoritma odrediti matricu dužina "najkraćih puteva". Traženi maksimalni broj
kutija je $abs(\min(ai\ j)) + 1,\ i\ j \in 1, .. , n$

Zadatak 11

Iz datog niza reči izdvojiti reči koje obrazuju najduži podniz reči takvih da svake dve susedne imaju zajednički segment od najmanje dva znaka. Na primer, 'NEVEN' i 'VENAC' mogu biti dve susedne reči.
Uputstvo:
Formirati matricu veze A u kojoj je $ai, j = -1$, ako se na i-tu reč može nadovezati j-ta reč, u protivnom $a\ i, j = \infty$ (*MAX*). Zatim, primenom Flojdovog algoritma odrediti matricu dužina "najkraćih puteva". Traženi maksimalni broj reči je $abs(\min(ai\ j)) + 1,\ i\ j \in 1, .. , n$

Zadatak 12

Inspektor Vasa je veoma iskusan u razresavanju krivicnih dela. Vasa rasvetljava krivicne dogadjaje tako sto desavanja koja prethodi dogadjaju predstavi kao skup uzrocno-posledicnih logickih iskaza oblika A=>B,

gdje su *A* i *B* događaji koji se često ponavljaju u detektivskim slučajevima. Logicki iskaz govori da je Vasa zaključio da ako se dogodi događaj *A* onda se nužno dogodio i događaj *B*. Iskaz često mogu da obrazuju i lance iskaza A=>B=>C, ali se **nikad neće pojaviti** kružni lanac iskaza (ciklus A=>B=>C=>...A), jer Vasa obavlja svoj posao pazljivo. Pomocnici na primer predstave Vasi skup događaja ***S*** = {S1, S2, S3, ... Sn} za koje postoje dokazi da su se ti dogadjaji dogodili. Vasa tada koristeći svoje metode razmisljanja odgovara koji su se sve događaji, uz one
navedene u ***S***, sasvim sigurno dogodili. Važno je znati da je Vasino znanje i iskustvo toliko veliko da su mu poznati **svi mogući uzroci svih mogućih događaja**. Dakle, ukoliko neka logicki iskaz u formi implikacije njemu nije poznat, tada taj iskaz nije ispravan. Ako, pak, Vasa za neki događaj **X** zna jedan ili više uzorka
i pomocnici skupe nepobitne dokaze za **X** tada je nužno da je barem jedan od mogućih uzroka koje Vasa zna zaista uzrokovao **X**. Dakle, nije moguće da se događaj koji ima

moguće uzroke dogodi na bilo koji drugi način osim kao posledica nekog prošlog događaja.

ULAZNI PODACI

U prvom redu ulaza nalaze se tri prirodna broja **D** (1 ≤ **D** ≤ 1000), broj različitih događaja za koje Vasa zna, **M** (1 ≤ **M** ≤ 100000) broj implikacija,
N (1 ≤ **N** ≤ **D**) broj dokaza koje su skupili pomocnici.
U sledećih **M** redova nalaze se po dva prirodna broja **A** i **B** (1 ≤ **A, B** ≤ **D**) koji označavaju implikaciju A=>B.
U sl
edećih **N** redova nalazi se po jedan prirodni broj **X** (1 ≤ **X** ≤ **D**) koji predstavlja događaje za koje pomocnici sigurno znaju da su se dogodili.

IZLAZNI PODACI

U prvi i jedini red ulaza potrebno je ispisati brojeve onih događaja koji su se sigurno dogodili. Događaji se mogu ispisati u bilo kom redosledu. Na izlazu je potrebno ispisati i one događaje za koje pomocnici već imaju nepobitne dokaze.

TEST PRIMER

Ulaz:

3 2 1
1 3
2 3
3

Izlaz:

3

Pojašnjenje primera: Događaj 1 implicira događaj 3, događaj 2 implicira događaj 3. Pomocnici su nesumnjivo dokazali događaj 3. Iako Vasa zna da je događaj 3 uzrokovan ili događajem 1 ili događajem 2 (nije moguće da se događaj 3 dogodio sam od sebe ili je uzrokovan nekim nepoznatim cinjenicama) on ne zna koji **tačno** od tih događaja ga je uzrokovao.

Ideja: Podelimo sad graf na tri tipa cvorova. Prvi tip su dokazani događaji. Za njih znamo da su se dogodili i oni trivijalno ulaze u rešenje. Drugi tip su svi događaji do kojih postoji put iz dokazanih događaja. Pokretanjem topoloskog sortiranja ili "flood fill" algoritma iz dokazanih događaja možemo prebrojati sve takve događaje i ubaciti ih u rešenje. Preostali su samo događaji koji se nalaze "pre" dokazanih događaja (imaju put koji vodi u dokazane događaje). Kako bismo odredili koji od njih ulazi u rešenje, uocimo da:
Za svaki događaj, pretpostavimo da ne može biti stvaran**.** Potom pokušavamo za što više događaja pokazati da su možda ostvareni.
Ako uspemo tako možda ostvariti sve dokazane događaje, zaustavimo dalju obradu. Ako ne uspemo, onda to znači da je pretpostavka nužno pogrešna i da se ispitivani događaj morao dogoditi.
Brzina ostvarenja te provere ključna je za ostvarivanje dobre vremenske slozenosti. Pogledajmo kako najbolje ostvariti tu proveru. Primetimo da postoji skup posebnih cvorova u grafu, koji imaju izlaznih grana, ali nemaju nijednu ulaznu granu. Nazovimo takve cvorove aktivatorima. Za svaki cvor možemo stvoriti skup aktivatora iz kojih postoji put do tog cvora. Ukoliko posmatramo dva cvora, X i Y, te se pitamo "može li cvor Y biti dokazan, a da se cvor X sigurno nije dogodio" vidimo da je odgovor na pitanje potvrdan ukoliko postoji barem jedan aktivator koji se nalazi u skupu aktivatora Y, a da u isto vreme nije prisutan u skupu aktivatora cvora X. Ovim možemo vrlo brzo odrediti odgovor na postavljeno pitanje. Drugo važno opažanje je da su aktivatori nekog vrha unija svih aktivatora neposrednih roditelja tog vrha.

Na prvi pogled čini se da je složenost ovakvog ispitivanja O(N^3) kao broj cvorova * broj cvorova * kardinalni broj skupa aktivatora cvorova. Međutim, pažljivija analiza poslednjeg cinioca nas vodi do konstantnog vremena i bolje slozenosti.

Tranzitivno zatvorenje

Svodimo na problem traženja svih najkraćih puteva.

Pravimo G' = (V, E') tako da se grani (u, v) dodeljuje težina

težina(u, v) = 1, (u, v) $\in$ E
0, inače

Ako u G postoji put od u do v, njegova dužina u G' je 0.

Algoritam

```
for (m : 1 .. n) {
    for (i = 1 .. n) {
        if (A[i, m]) {
            for (j = 1 .. n) {
                if (A[m, j]) {
                    A[i, j] = true;
                }
            }
        }
    }
}
```

Zadaci

Zadatak 1

Imamo tri ugnežđene petlje u prethodnom algoritmu. Spoljašnja bira kolonu, srednja vrstu, a unutrašnja obrađuje vrstu. Da li algoritam radi ako zamenimo redosled prve dve petlje?